WL-TR-91-3056

AD-A242 374

FEM-X USER'S GUIDE

Warren C. Gibson
CSA Engineering, Inc.
560 San Antonio Road
Suite 101
Palo Alto, CA 94306

23 August 1991

Final Report for Period Mar 89 - Dec 90

DTIC
ELECTE
NOV 13 1991
S B D

Approved for public release; distribution unlimited

91-13259

FLIGHT DYNAMICS DIRECTORATE
WRIGHT LABORATORY
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433-6553

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely Government-related procurement, the United States Government incurs no responsibility or any obligation whatsoever. The fact that the goverment may have formulated or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication, or otherwise in any manner construed, as licensing the holder, or any other person or corporation; or as conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

This report is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.

STEPHEN M. RASMUSSEN, 1Lt, USAF
Project Engineer
Design & Analysis Methods Group

NELSON D. WOLF
Technical Manager
Design & Analysis Methods Group
Analysis & Optimization Branch

FOR THE COMMANDER

DAVID K. MILLER, Lt Col, USAF
Chief, Analysis & Optimization Branch
Structures Division

If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify WL/FIBRA, WPAFB, OH 45433-6553 to help us maintain a current mailing list.

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
23 August 91

3. REPORT TYPE AND DATES COVERED
FINAL MAR 89 - DEC 90

4. TITLE AND SUBTITLE
FEM-X User's Guide

5. FUNDING NUMBERS
PR 3005
TA 40
WU 42
PE 62201F
C F33615-89-C-3205

6. AUTHOR(S)
Dr Warren C. Gibson

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
CSA Engineering, Inc.
560 San Antonio Road
Suite 101
Palo Alto CA 94306

8. PERFORMING ORGANIZATION REPORT NUMBER
91-01-03

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Lt Stephen Rasmussen (513) 255-6992
Flight Dynamics Directorate
Wright Laboratory
WL/FIBRA
Wright-Patterson Air Force Base OH 45433-6553

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
WL-TR-91-3056

11. SUPPLEMENTARY NOTES
This is a Small Business Innovative Research Program, Phase II report.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for Public Release; Distribution Unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This document is a User's Guide for FEM-X, a database management system for finite element models. The Guide describes the organization of finite element model data and descriptive text in FEM-X, and describes briefly the operations of model data entry and text data entry, and extraction and translation of model data. FEM-X is a window-based product designed to run on engineering workstations under the X Window System. The graphical interface makes operation of FEM-X intuitive, with on-line help available at each stage. Discussions of CADDB, the database system used for FEM-X, and its query software, ICE, are also provided.

14. SUBJECT TERMS
Finite Element Models, Finite Element Models Data Base, Storage of Finite Element Models

15. NUMBER OF PAGES
39

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED

FOREWORD

This report is part of the Phase II SBIR effort conducted by CSA Engineering, Inc. under Air Force contract F33615-89-C-3205, entitled "Finite Element Models for the Supportability of USAF Aircraft." CSA was assisted by its subcontractors, Aerospace Structures, Inc. and Applied Technology, Inc. The report covers work performed during the period 9 Mar 89 through 31 Dec 90. Lt. Steven Rasmussen was the Air Force project monitor for WL/FIBR.

Prepared by: ______________________

Warren C. Gibson, Ph.D.
Principal Engineer

Accession For
NTIS GRA&I
A-1

Contents

List of Figures

1. Introduction

This document is a User's Guide for FEM-X, a database management system for finite element analysis. FEM-X was developed for Wright Laboratory, Flight Dynamics Directorate (WL/FIBR), Wright-Patterson Air Force Base, Ohio, by CSA Engineering, Inc. under a Phase II SBIR effort. The final report for this effort, "Preservation and Utilization of Finite Element Models of USAF Aircraft Structures," [1] accompanies this User's Guide. Section 6 of that report presents some background information about FEM-X, including anticipated uses for the program. The name FEM-X is taken from "Finite Element Models" and "the X Window System."

CSA's contract calls for delivery of "prototype" DBMS software, and the version of FEM-X delivered with this report is indeed prototype software. There has been no "beta" testing of this software, and therefore CSA does not recommend distribution of FEM-X until such testing has been completed.

FEM-X is a window-based product designed to run on engineering workstations under the X Window System. It was designed to guide the user at every step through intuitive menus and a context-specific "help" button. It is not expected that users will have to have this Guide available at all times when using FEM-X. Therefore the Guide focuses more on the concepts used in the software and is intended to be read by new users and consulted occasionally by experienced users. New users are advised to try the program after only a cursory reading of this Guide. In most cases, it is much easier to learn by doing because the functions of FEM-X were designed to be intuitive and simple. Often a verbal description of a simple function can be rather lengthy.

Appendix A provides a brief introduction to the X Window System for users who are new to X. Appendix B is a brief introduction to CADDB (the database software that FEM-X uses) and ICE, a query program for CADDB databases.

FEM-X is driven by two sets of tables which are accessible to advanced users. Advanced users may want to modify these tables, when, for example, a new release of NASTRAN appears. Sections 10 and 11 describe these tables, which govern entry of bulk data into FEM-X and translation of bulk data between software formats. Section 12 briefly describes the function of FEM-X in terms of data flow and control flow. Appendix C gives a list of files that are delivered with FEM-X.

2. The Hierarchy of Systems, Components, Versions, and Variations

Briefly, a "system" is an entire aircraft, and a "component" is a portion of the aircraft that is modeled by a complete finite element model.[1] (It is unusual to have a single model of an entire aircraft, but FEM-X can handle this case by simply defining a system with a single component.) A "version" represents a modification of a particular component, such as an alternate type of construction, modified stores, etc. A variation represents a minor change to a model, typically for purposes of mesh refinement, alternate selection of elements, etc. It is the user's responsibility to designate a particular modification as a version or variation. Note that a variation can apply to either a component model or a version.

The hierarchy of systems, components, versions, and variations is shown in Figure 1. The user interface windows in FEM-X are arranged in a corresponding hierarchy. Thus, from the top level, the user selects a particular system (or creates a new one), and can move down the hierarchy to components, versions, and variations, and back up again. There is only one active window at any one time (with the exception of "dialog boxes" and "help" windows discussed below). In order to

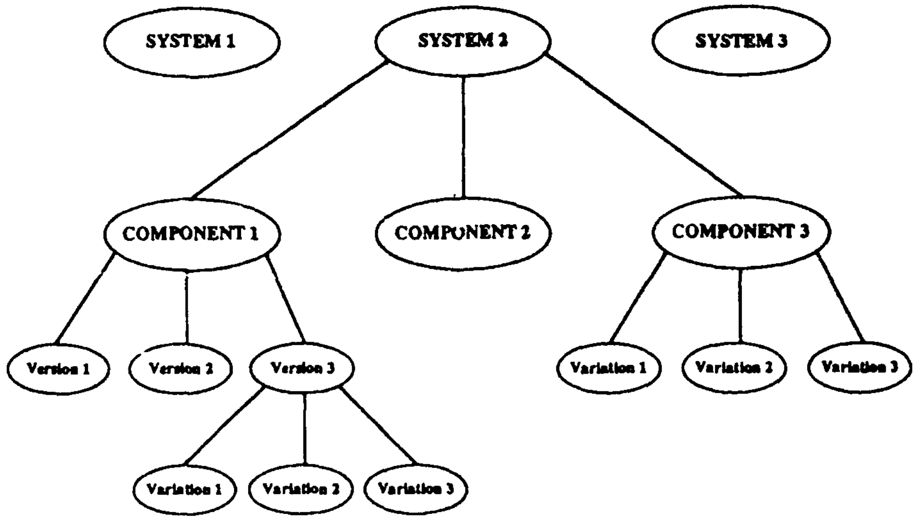


Figure 1: Hierarchy of Information in FEM-X

[1]These concepts are discussed in more detail in section 6 of the accompanying report.

move to a component in a different system, for example, the user must move back up the hierarchy, select the new system, then select the desired component. It is possible to move up more than one level with a single mouse pick, but downward movements must be made one level at a time.

3. Screen Layout

A typical FEM-X window is shown in Figure 2. All main FEM-X interface windows share certain common attributes. At the top of each window are three pulldown menus, labeled "Control," "Display," and "Manage." The Control menu is used to move up the hierarchy or to exit from FEM-X. From the current level in the hierarchy, the user may move up one level, or as many levels as desired. When such a move is made, any activity that had been in progress relative to the current window is terminated. In particular, if the user had been entering data through a window and chose to jump back up to a higher level without finishing the data entry, all information that had been typed in that window is lost. The "Display" menu offers the opportunity to move down the hierarchy, or to display a "Help" window. The "Manage" button offers the opportunity to enter a new system, component, etc., or to delete an old one. The present prototype version does not offer any protection such as a password, to restrict these functions to certain users.

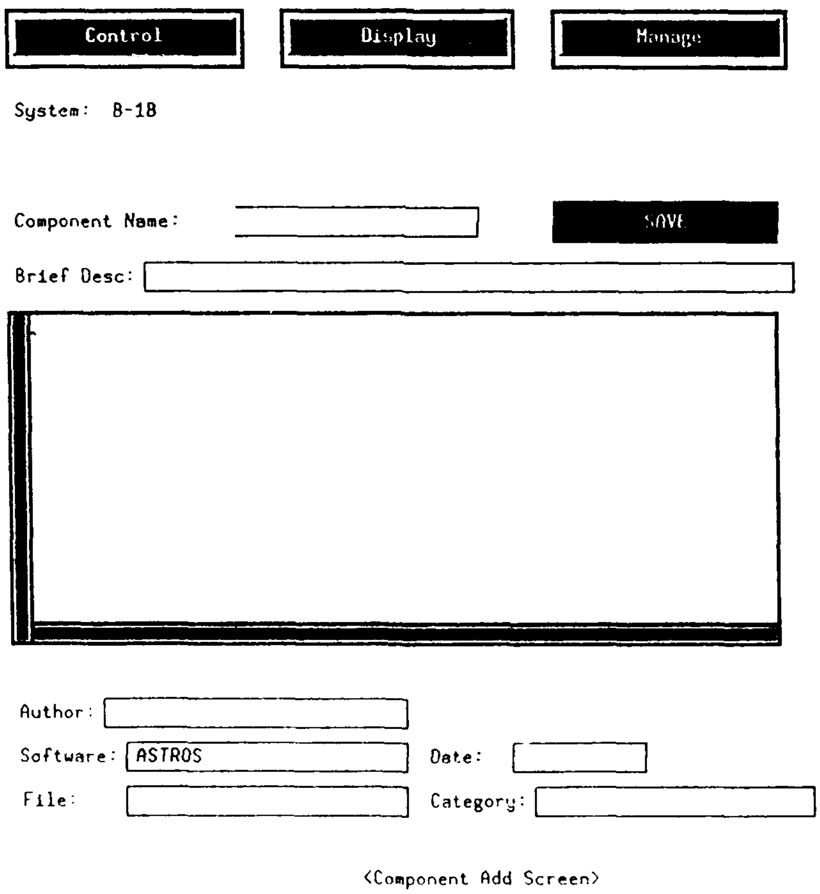


Figure 2: A Typical FEM-X Window

The pull-down menus do not work exactly like the pull-downs familiar to Macintosh computer users. In Macintosh windows, one picks the pull-down menu title and then, while holding the mouse button moves down to the desired action and releases the button. In FEM-X, it is not necessary to hold the button, but it must be clicked a second time on the desired entry. If the pointer is moved out of the pull-down area while all mouse buttons are up, the pull-down disappears.

4. Entry of Text Data

The window shown in Figure 2 is an example of a window in which the user is to enter information. This particular screen appears when users select "Add Component" under the "Manage" pulldown in the "Component List" screen. There are several boxes in which short text strings are to be typed. Text is entered here by placing the mouse pointer in the appropriate box and then typing the text. Errors are corrected by pressing the DELETE key. Macintosh users will not see the familiar "I-beam" cursor which is used when text is entered on that computer. The large sub-window labelled "Detailed description" is based on a software tool called an "Athena Text Widget." Such windows are used throughout FEM-X whenever more than one line is to be entered. The user is offered a complete set of editing functions, corresponding to the `xedit` text editor that is supplied with the X Window System (see Appendix A for a summary of `xedit`). Of particular importance here is the ability to paste selections that were cut out of another window. For example, if a descriptive document were available in an ASCII file, it could be displayed in another window on the same screen with FEM-X. Portions of the text in from this window could be selected and inserted into the FEM-X database using the standard cut-and-paste facility of the X Window System. Cut-and-paste is a four-step process which is easier to do than to describe:

1. Move the mouse to the front of the text and depress and hold the left mouse button.
2. "Drag" the pointer to the end of the text (i.e., move the pointer while holding down the left mouse button), then release the button. The selected text will appear in reverse video.
3. Move the pointer to the desired location in the FEM-X window, and click the left mouse button. This establishes the insert position in the text, and the carrot cursor appears there.
4. Depress the middle mouse button momentarily to insert the text.

Note that this will work even if the two windows originate from programs running on different computers. If the text to be cut from an `xedit` window, and the text extends beyond what currently appears in the window, one can scroll forward or backward using the ctrl-v or alt-v commands or by manipulating the scroll bar. The selection can then be extended by clicking the right mouse button. An alternative to the cut-and-paste approach is the insert file approach. One can insert an entire file at the current insertion point by pressing alt-i. See Appendix A for more information.

FEM-X allows users to enter rather lengthy detailed descriptions if they so choose. However, this facility is not intended to hold dozens of pages of text. The prototype version of FEM-X has a built-in limit of 50,000 characters, or roughly 20 pages of text. This limit can easily be changed inside the FEM-X software.

5. Entry of Finite Element Model Data

Models that are entered into FEM-X must conform to one of three data formats: ASTROS, COSMIC NASTRAN, or MSC/NASTRAN. For more information on these formats, consult the reference manuals for these programs [2] [4] [5]. When the user adds a new component, the name of the ASCII input file must be given. FEM-X then splits this file into two parts: a "preface" file and a bulk data file. The bulk data is always entered into the FEM-X database, and the preface may be kept under a user-specified name or discarded.

Prefaces consist of preliminary information, (executive control, case control, etc.). Prefaces are stored separately from bulk data and are also tagged as having either ASTROS, COSMIC NASTRAN, or MSC/NASTRAN format. FEM-X does not perform any kind of checking to validate preface data, however. The contents of prefaces for the various finite element packages are as follows:

ASTROS

DEBUG packet	Debug output control
MAPOL packet	MAPOL matrix language source code
SOLUTION packet	selections of solution type, loads, support conditions, output requests, solution methods, etc.

COSMIC NASTRAN

NASTRAN line	Miscellaneous operating parameters
Executive control	Choice of solution type, etc.
Substructure control	Control of substructuring operations
Case control	Selection of loads, support conditions, output requests, solution methods, etc.

MSC/NASTRAN (version 65 and earlier)

NASTRAN line	Miscellaneous operating parameters
Executive control	Choice of solution type, etc.
Case control	Selection of loads, support conditions, output requests, solution methods, etc.

MSC/NASTRAN (version 66 and later)

NASTRAN line	Miscellaneous operating parameters
File management section	Database control
Executive control	Choice of solution type, etc.
Case control	Selection of loads, support conditions, output requests, solution methods, etc.

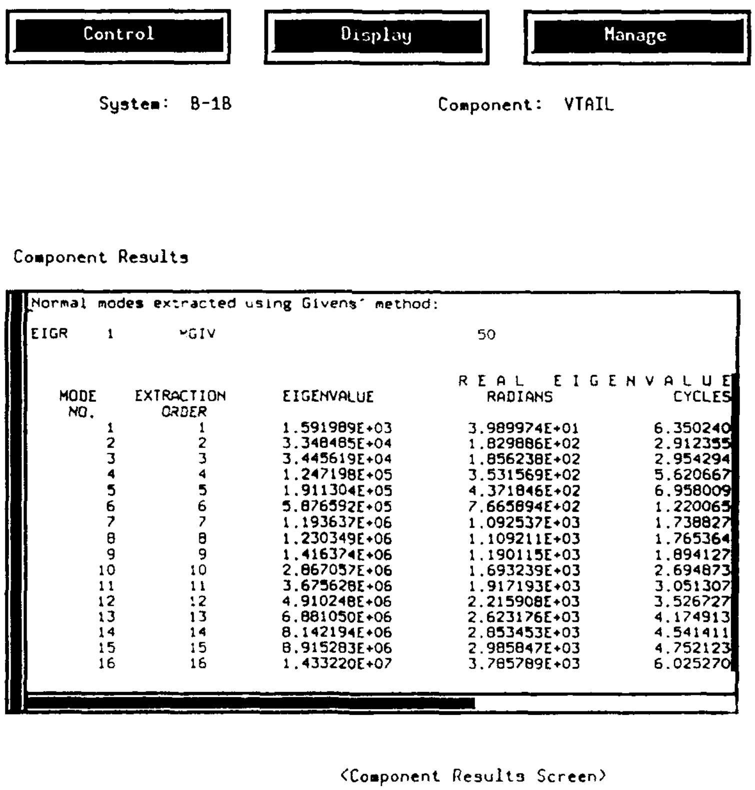
Bulk data is then read into the database in CADDB format (see Section 10 and Appendix A). At this time, some cursory checking is performed on the bulk data: existence of required data, proper data type, etc. There are some restrictions on the format of bulk data entries, as follows:

1. Continuation lines must immediately follow their parents. This is an ASTROS requirement, and although it is not a requirement for NASTRAN, NASTRAN users overwhelmingly observe this rule as a convention.
2. No automatic data-generation symbols, as provided in COSMIC NASTRAN and MSC/NASTRAN, are allowed (using special symbols =, $, (), etc.).
3. No mesh generation commands for MSC's MSGMESH mesh generation option are allowed. (Files which violate any of the first three restrictions should be submitted to NASTRAN with the ECHO=PUNCH option to generate an acceptable bulk data file.)
4. Dummy elements are not supported. The format of the associated records (CDUMi, PDUMi, etc.) is determined by the author of the dummy element. If support for a particular dummy element were desired, the SYSGEN template and relation files could be modified to support it (refer to Section 10 for more information).
5. Axisymmetric elements are excluded. These are considered obsolete elements, certainly unsuitable for aircraft analysis. However, they could be added to the SYSGEN files if needed.
6. Large GENEL, DMI, or DTI records may be handled inefficiently (specifically, the associated bulk data relation will have one record for each item in the bulk data record).

6. References and Results

FEM-X provides text windows in which users may enter information under the headings "References" and "Results." The user is entirely free to enter any desired text in these windows. However, it is anticipated that "References" will be used for information like the names of reports related to the model, memos resulting from telephone conversations, etc. A "Results" window appears in Figure 3.

This feature is expected to be used for summaries of the results produced when a particular model is run. It should also include the analyst's comments about the validity and limitations of the model. As with other text windows, the select-and-paste facility of the X Window System will make it convenient to copy information from other documents or from NASTRAN output files.


Control Display Manage

System: B-1B Component: VTAIL

Component Results

Normal modes extracted using Givens' method:

EIGR 1 MGIV 50

MODE NO.	EXTRACTION ORDER	EIGENVALUE	REAL EIGENVALUE RADIANS	CYCLES
1	1	1.591989E+03	3.989974E+01	6.350240
2	2	3.348485E+04	1.829886E+02	2.912355
3	3	3.445619E+04	1.856238E+02	2.954294
4	4	1.247198E+05	3.531569E+02	5.620667
5	5	1.911304E+05	4.371846E+02	6.958009
6	6	5.876592E+05	7.665894E+02	1.220065
7	7	1.193637E+06	1.092537E+03	1.738827
8	8	1.230349E+06	1.109211E+03	1.765364
9	9	1.416374E+06	1.190115E+03	1.894127
10	10	2.867057E+06	1.693239E+03	2.694873
11	11	3.675628E+06	1.917193E+03	3.051307
12	12	4.910248E+06	2.215908E+03	3.526727
13	13	6.881050E+06	2.623176E+03	4.174913
14	14	8.142194E+06	2.853453E+03	4.541411
15	15	8.915283E+06	2.985847E+03	4.752123
16	16	1.433220E+07	3.785789E+03	6.025270

<Component Results Screen>

Figure 3: "Results" Window.

7. Scripts

FEM-X allows users to create and store scripts, which are intended primarily to define subsets of bulk data records. A subset is simply one or more bulk data records selected from a particular bulk data file according to criteria defined by the user. Scripts are expected to be useful primarily to analysts who are engaged in the process of modifying a model and need to identify elements, grids, etc., that lie in a certain region of the model, have certain properties, represent certain types of material, etc.

Users write scripts in CQL (CADDB Query Language), the language that is implemented in ICE [3]. ICE offers a wide variety of selection commands that may be used to establish set membership criteria. The scripts are entered in the user's FEM-X database along with descriptive commentary. It is incumbent on the user to write syntactically correct CQL code. FEM-X does not check the code, but merely sends it to ICE for execution and displays the result of that execution. The user must use the `SET INTERFACE` command in ICE in conjunction with the `INTERFACE FORMAT` command in order to create an ASCII file that is suitable for inclusion in a bulk data deck to be submitted to NASTRAN or ASTROS.

An example will help clarify the foregoing discussion. Assume that a user is working with a model of a wing and wants to refine the model in the region around an engine mount for the purpose of more accurate stress calculations. The user wants to find all the CQUAD4 and CTRIA3 element records on the top and bottom wing surfaces near the engine mount. The engine mount is located along a vertical plane at Y=191.2. All CQUAD4 and CTRIA3 elements having any of their grid points within 14" of this plane are to be included. The following ICE script, with lines numbered for discussion purposes, accomplishes this.

```
1       CREATE VIEW ENGINEGRIDS AS
          SELECT ID,X,Y,Z FROM GRID WHERE
          ABS(Y-191.2) <= 14.0;
2       SET INTERFACE TO 'ENGINE.GRID';
3       INTERFACE ON;
4       INTERFACE FORMAT '(''GRID'',I12,8X,3F8.4)';
5       SELECT ID,X,Y,Z FROM ENGINEGRIDS;
6       SET INTERFACE TO 'ENGINE.CQUAD4';
7       INTERFACE FORMAT '(''CQUAD4'',T9,6I8)';
8       SELECT EID,PID,G1,G2,G3,G4 FROM CQUAD4 WHERE
          G1 IN (SELECT ID FROM ENGINEGRIDS) OR
          G2 IN (SELECT ID FROM ENGINEGRIDS) OR
          G3 IN (SELECT ID FROM ENGINEGRIDS) OR
          G4 IN (SELECT ID FROM ENGINEGRIDS);
9       SET INTERFACE TO 'ENGINE.CTRIA3';
10      INTERFACE FORMAT '(''CTRIA3'',T9,5I8)';
11      SELECT EID,PID,G1,G2,G3 FROM CTRIA3 WHERE
```

```
G1 IN (SELECT ID FROM ENGINEGRIDS) OR
G2 IN (SELECT ID FROM ENGINEGRIDS) OR
G3 IN (SELECT ID FROM ENGINEGRIDS);
```

Line 1 selects all GRIDs within the specified region and saves them in a VIEW.[1] Line 2 indicates a file to receive the GRID record images and line 4 specifies the proper NASTRAN GRID format. Line 5 selects all entries from the ENGINEGRIDS relation and causes them to be written to the interface file. Line 6 opens a new file to receive CQUAD4 records,[2] line 7 specifies the format, and line 8 writes the CQUAD4 elements. Similarly, lines 9 through 11 create CTRIA3 data. The user would use these files in refining the model, using a text editor or a finite-element pre-processor.

[1]The VIEW command performs the same function as the SELECT command, except in addition it saves the results under the name (ENGINEGRIDS) given by the user.

[2]There is a bug in ICE: whenever a new interface format is specified, any existing information that had been written to an interface file is lost, hence the three separate file specifications on lines 2, 6, and 9. Also, ICE does not allow comment lines in script files, which would be useful.

8. Extraction and Translation of Finite Element Model Data

FEM-X allows users to create input file for NASTRAN or ASTROS, consisting of a preface (executive control, case control, etc.) followed by a bulk data deck. This function is invoked by selecting a particular component, and a version or variation if desired, then picking "Extract" in the "Manage" pull-down. The window that appears is shown in Figure 4. The user may choose a preface to include with the file. To choose a preface, the user may choose the window that appears is shown in Figure 4.

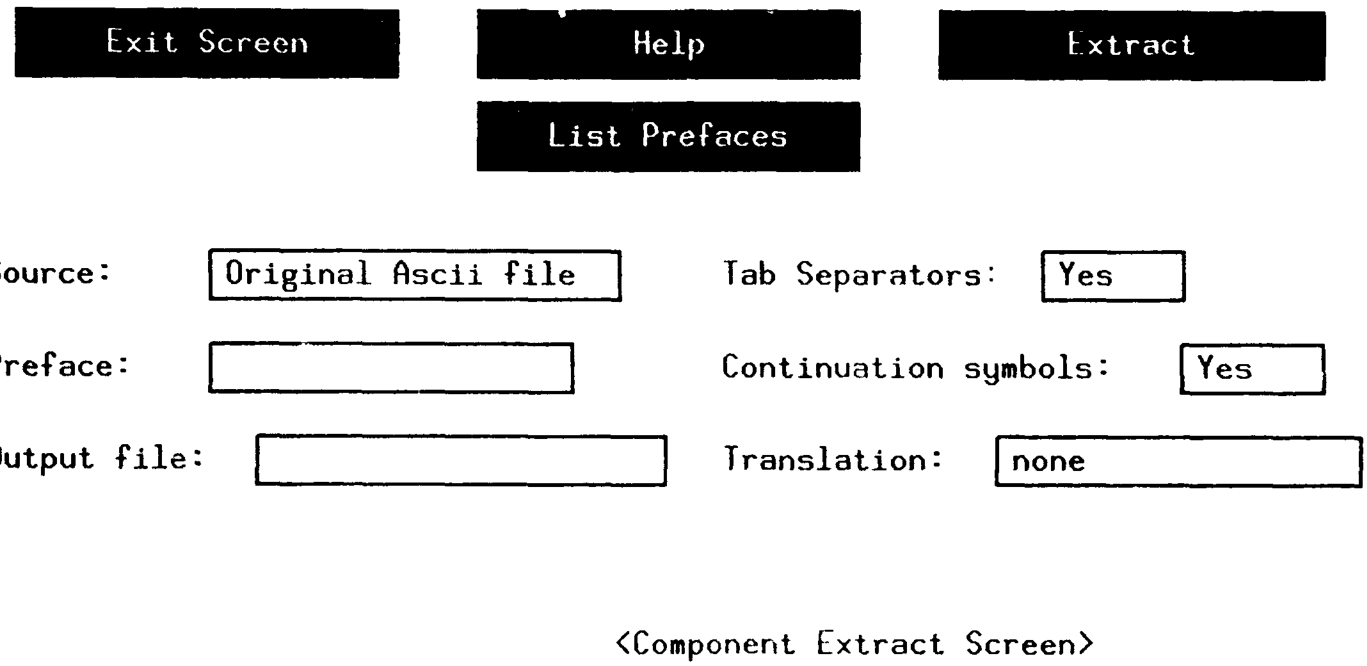


Figure 4: "Extract" Window.

The user may choose a preface to include with the file. To choose a preface, the user may type its name in the field provided, or pick the "List Prefaces" button. This button causes a scrolling window to appear, in which the names of available prefaces are listed. A preface may be selected from this list by picking it with the left mouse button. There are two options regarding the source of the bulk data. Although the CADDB database is the preferred choice, users have the option of selecting the original ASCII file instead. This choice is discussed in Section 6.3.1.2 of the main report. The choice is made with a button that acts as a toggle, i.e., when picked it switches between "Original ASCII file" and "CADDB database." If

"CADDB database" is selected, two formatting options are offered: tab separators and continuation symbols. MSC/NASTRAN allows tab characters between fields instead of blanks, and also allows omission of continuation symbols in fields 10 and 1. These are both Yes/No toggles. The other field offered is "Translation," which is used to request translation of the bulk data to either of the other two formats (e.g., ASTROS data may be translated to either COSMIC NASTRAN or MSC/NASTRAN format).

It is not possible to guarantee a one-to-one translation of all bulk data records, because each of the three finite element codes supports features that are not supported in the other, or supports them differently. But because the vast majority of most bulk data files consist of common records like GRID, CQUAD4, CTRIA3, SPC, FORCE, etc., which *are* common to all three codes, translation is a useful feature even though it may sometimes leave the user with difficult decisions. The translator generates a file in which it reports all the translation anomalies that it encountered. Untranslatable records are simply omitted from the translated file, and are noted in the report file. Where assumptions can be made (as in translating from NASTRAN to ASTROS, when a SET ID must be provided for ASET records), they are made and noted.

9. FEM-X User Environment

FEM-X runs on workstations that support Unix and the X Window System. For more information on the Unix concepts discussed here, consult Unix reference documents, the **man** pages available on most systems, or a resident expert.

Unix users may choose the command language under which they operate, the two most common being the Bourne shell and the C-shell. The following discussion assumes the C-shell is used.

An environment variable called FEMX must be defined, and must indicate the directory where FEM-X is installed, e.g.,

```
setenv FEMX /home/femx
```

The actual directory name will vary among installations. An alias should be then be defined in order to run FEM-X:

```
alias FEMX $FEMX/FEMX
```

The command line for executing FEM-X is

```
FEMX <path> <master>
```

where `<master>` is the name of the master database and `<path>` is its path. FEM-X is designed to work from a single master database. However, there is nothing to prevent users from establishing a private master database. If the master database specified on the command line does not exist, a warning is issued and a new, empty database is created. Assuming a number of users wish to access a central database called `master` and that this database is located in the same directory as the software, the command line for executing FEM-X would be

```
FEMX $FEMX master
```

Note that CADDB is not a true multi-user database. It is possible for multiple users to access the same database simultaneously, provided the Unix file permissions allow this, but the results will not be right. The second user to open a database will see a warning that the database was not properly closed in the previous run, but will be able to proceed. However, changes made by the first user will not be seen by the second user, and there is a risk that the whole database will be destroyed. In order to avoid such conflicts, FEM-X simply locks a master database whenever a user begins using it, preventing other users from running FEM-X on that database until the first user is finished.

FEM-X creates a number of directories as it works. If a user creates a new master database called `master`, FEM-X creates a subdirectory called `master.femx` in the directory given by `<path>`, containing CADDB database files called FEMX.D01 and FEMX.IDX, i.e., the full file names would be

```
<path>/<master>.femx/FEMX.D01
<path>/<master>.femx/FEMX.IDX
```

When users create new systems (using the "Add System" option under "Manage"), a subdirectory is created in `<path>/<master>.femx`. For example, if a new system called **B-1B** is created, a directory `./B-1B.sys` is created (the leading period stands for the ectory in Unix). Initially, that directory is empty. When a component is added, a subdirectory is created within the system directory, using the component name and the suffix **.cmp**. Thus, for example, if a wing carry-through model is added under the name WINGCTHR, a directory called WINGCTHR.cmp is created. Three files are created in this directory. The bulk data (without preface data) appear in **bulk** in ASCII format and in a database called BULK in CADDB format. Thus, in the example, the following files would appear:

```
<path>/<master>.femx/B-1B.sys/WINGCTHR.cmp/bulk
<path>/<master>.femx/B-1B.sys/WINGCTHR.cmp/BULK.D01
<path>/<master>.femx/B-1B.sys/WINGCTHR.cmp/BULK.IDX
```

Similarly, creation of versions and variations result in creation of directories with suffixes **.ver** and **.var** containing the same types of files. When users create new components, the bulk data is assumed to reside in the directory that was current when FEM-X was started, unless an explicit path is typed along with the bulk data file name.

In order to use FEM-X, users must first start the X server software on their local workstation or X Terminal, and then type the FEM-X command line from an **xterm** window on their screen. If FEM-X is being executed from a remote computer, the user must log in to that computer first. More information on the X Window System, including the initialization command xinit and the initialization resource file **.xinitrc**, may be found in Appendix A.

10. The FEM-X SYSGEN Process

This section for advanced users who may need to modify the action of FEM-X, perhaps in response to a new release of NASTRAN, and to other interested users. Some background information on ASTROS is necessary before explaining the FEM-X SYSGEN process.

ASTROS [2] is an engineering analysis and optimization system based on the finite element method. ASTROS was developed for the Wright Laboratories Flight Dynamics Directorate, and is distributed without charge to qualified users. The developers of ASTROS saw fit to create their own database management system to support ASTROS, having concluded that no existing DBMS package available to them would suit their purposes. CADDB (Computed-Aided Design Data Base) is the name of the system they developed. It was designed to fill the needs of a modern finite element package and therefore stores data in three formats: relations (i.e., tables of numbers or short character strings), large sparse rectangular matrices, and unstructured entities (for "scratch" purposes).

Every time ASTROS runs, it either creates a new database or accesses an old one. In addition, there is a "system database" that is generated when ASTROS is installed. Thus ASTROS uses CADDB not only for user data but to store some of its own internal structures as well.

ASTROS is driven by a matrix-oriented language called MAPOL. There is a "standard sequence" of several hundred lines of MAPOL code that provides all the engineering features of ASTROS. Users are allowed to modify this sequence or substitute their own.

User input is submitted to ASTROS through ASCII files which are patterned after NASTRAN's input. Following some preliminary information, the actual engineering data appears in "bulk data" format, a sequence of logical records that follow specified formats.

In order to maximize the utility of ASTROS, its developers elected to "hardwire" as little as possible of its features. Instead, many features are coded in ASCII files which are accessible to users. These files drive a process called SYSGEN which results in the creation of the system database, which subsequently drives all ASTROS executions. Five kinds of information are coded in these files:

1. Descriptions of all possible bulk data records.[1] For each record type, this file includes the record name, the individual field names, field data types (real, integer, character), error checking criteria, and the correspondence of records and fields with relations and attributes in the database.

2. Descriptions of all bulk data relations in the database. These include relation names, attribute names, and attribute types.

[1] sometimes called "cards" for historical reasons

3. Source code for the standard MAPOL sequence.

4. Definitions of all MAPOL-callable modules (name, number, and type of arguments).

5. Error messages.

FEM-X takes advantage of the ASTROS SYSGEN process to read bulk data files into CADDB relations. The files may be in ASTROS, COSMIC NASTRAN, or MSC/NASTRAN format. There are three corresponding "rump" versions of ASTROS which were generated for this purpose. Each of them uses a MAPOL program which consists of little more than a call to module IFP, which reads bulk data, checks it, and writes it into CADDB relations. For ASTROS files, the standard ASTROS SYSGEN files are used, with the exception of the MAPOL and error message files, which are truncated. For COSMIC NASTRAN and MSC/NASTRAN, it was necessary to create new files which describe the bulk data records and the CADDB bulk data relations. Running SYSGEN against these files produces three executable "rump" versions of ASTROS and three corresponding databases. Running any of these rump versions results in an ASTROS-like output file which echoes bulk data and issues error messages, and a user database filled with bulk data relations.

The actual process of reading a bulk data file into a CADDB database consists of the following steps. These steps are coded in a shell script file called **read_dat** which is executed by the main FEM-X code when a new component, version, or variation is added.

1. Split the user's input file into a preface (executive control, case control, solution control, etc.) and the actual bulk data.

2. For MSC/NASTRAN bulk data decks, change tab separators to blanks and insert continuation symbols if the continuation fields were left blank in the source file.

3. Attach an ASSIGN statement and a dummy solution packet to the front of the user's bulk data.

4. Run the proper "rump" version of ASTROS (called READAST, READCOS, or READMSC).

5. Search the output file from READAST, READCOS, or READMSC for error messages.

6. Add a special relation called CARDDEF to the user's database. This relation contains the names of all possible bulk data fields, and it is used in extracting bulk data.

7. Return a status code to FEM-X.

11. The Translation Process

It is not possible to effect a one-to-one translation of model data between finite element codes simply because different codes support different features. Nevertheless, there is enough commonality among the three supported codes (ASTROS, COSMIC NASTRAN, and MSC/NASTRAN), to warrant development of a translator. Besides, most aircraft models use few or none of the features which are not common to all three codes.

The following policies were adopted in developing the FEM-X translator:

1. Where the object software[1] does not support a particular record type that is present in the source file, bring that fact to the user's attention, leaving the user with a choice of whether to make a manual substitution of another record type that he considers suitable, omit the offending data, or abandon the translation.

2. Similarly, where a particular field type within a record has no equivalent in the object software, omit the untranslatable data and leave the decision to the user.

3. Where the object software record requires data that are not present in the source data, provide a default value, warn the user, and proceed. As an example, the ASET record in ASTROS requires a set ID, but COSMIC and MSC/NASTRAN do not. In this case, the translator assumes a value of 1 and proceeds.

4. Provide for different translation actions as a function of the value in a certain record of the input data, where necessary. As an example, consider the eigenvalue data record, EIGR. All the codes have two methods in common: INV and GIV. COSMIC NASTRAN and MSC/NASTRAN each provide additional methods that are not directly supported by the others. When translating from MSC/NASTRAN to COSMIC NASTRAN, method types INV and SINV are each translated to INV, while GIV, MGIV, HOU, and MHOU are all translated to GIV. In the reverse direction, the COSMIC DET, UDET, and UINV methods are all translated to INV while FEER is translated to GIV. The MSC/NASTRAN EIGRL record, used to request the Lanczos method, is translated to an EIGR card with the GIV method.

5. Provide for automatic insertion of remarks when warranted by special circumstances.

6. Ignore minor differences in behavior that may result from translation. Although the QUAD4 elements in COSMIC NASTRAN and ASTROS are slightly different from those in MSC/NASTRAN, these differences are not noted.

[1] i.e., the software format to which the data are being translated

Translation difficulties should not be overestimated. While it is true that the analyst will need to inspect the translation results in all cases, it is likely that for typical aircraft models, only a few lines of data will have to be changed. A typical modern model of an aircraft component might have the following distribution of records:

GRID	45%
CQUAD4	40%
CTRIA3	5%
CBAR	5%
PSHELL, PBAR, and MAT1	2%
SPC	2%
miscellaneous	1%

Only among the last 1% would any difficulties arise.

Implementation of all the translation possibilities in Fortran code would result in an enormous tangle of conditional clauses which would be virtually impossible to maintain. Instead, a driver file called TRANSNAS.DRV was devised to direct the translation process. The goal was to make this file easy to read, understand, and modify with a text editor. The following circumstances may require editing TRANSNAS.DRV:

1. Release of a new version of ASTROS, COSMIC NASTRAN, or MSC/NASTRAN. The delivered version supports ASTROS level 4.0, COSMIC NASTRAN June 1986, and MSC/NASTRAN version 65.

2. User preferences that differ from the developer's on matters that are judgement calls.

3. Errors in the delivered version. The developers did not have access to COSMIC NASTRAN, only the users manual, and were therefore somewhat vulnerable to misinterpretation of COSMIC NASTRAN data.

The first several lines of TRANSLATE.DRV appear in Figure 5.

The actual data used by the translator appears in three columns and is preceded by a number of comments which describe the format of the file. If, for example, the software is translating from ASTROS to COSMIC NASTRAN format, it scans the first column (for ASTROS) for each record in the source file. If there is no corresponding entry in the second column (COSMIC NASTRAN), the input record is skipped and a warning message is written on the report file. As the figure shows, the ACTIVE card is a mesh generation card which is ignored by the translator; hence the exclamation mark. The AEFACT card is identical in all three codes, and is open-ended, as indicated by "etc." The AELIST and AESTAT cards of MSC/NASTRAN have no counterpart in the other codes. The AERO card is the same in COSMIC and

```
Version 1.  Oct '89.  W. Gibson, CSA Engineering, Inc.
Version 1a. Feb '90.  W. Gibson -- add "p" and "a" qualifiers
Version 1b. Mar '90.  W. Gibson -- add more cards, fix errors

This is a driver file for use with the TRANSNAS translator.  Knowledgable
users can modify the translator's functions by editing this file. Any code usin
the bulk data style of data can be translated.  A new code may be added, or new
fields within an existing code can be added.

Currently ASTROS, COSMIC NASTRAN, and MSC/NASTRAN are supported.
Translation can be done between any pair of codes.  The "source" code
is the name of the program corresponding to the input bulk data, and the
"destination" code is the name of the program for which a translated
bulk data file is to be written.

Users familiar with NASTRAN should understand the form of this file simply
by looking through it.  After browsing, note the following rules:

RULES:
 1. There is one column for each supported code, each column 16 characters wide
 2. Tab characters may be used to align column data (tabs assumed set at
    characters 1,9,17,...).
 3. The names of the supported codes appear between rows of = signs at the
    head of the file, followed by the appropriate version identifier.
 4. Names of bulk data cards appear following blank lines.  The appearance
    of various card names on the same line indicates that they are equivalent
    and will be translated to each other.  Where a particular program has
    no equivalent, the word "unknown" appears.
 5. Although this file was originally set up with card names appearing in
    alphabetical order, there is no requirement that they be ordered.
 6. The * character following a card name indicates an exception.  In this
    case, the indicated name will be used only when translating TO the code
    corresponding to the particular column.
    card has no equivalent, the word "unknown" is coded in the appropriate
    column.
 7. The ! character following a card name indicates that it is a mesh
    generation card (e.g., an MSGMESH card for MSC/NASTRAN).  This raises
    a flag indicating the need to play the model thru NASTRAN to create
    expanded bulk data cards.
 8. A number in parentheses following a card name is an integer which
    points to a remark.  Remarks are coded at the end of this file
    (see rules  ).  The indicated remarks will be issued to the user when
    the indicated card is encountered in the source file.
 9. Following each card name is a list of field numbers and field names,
    separated by dashes.  Fields that appear on the same line are translated
    to each other.  Where a particular code provides no equivalent, a blank
    space appears.
10. "n" preceding a field number means that when translating TO this
    code it is necessary to place the data on a new copy of the card.
11. "c" preceding a field number means that when translating TO this
    code it is necessary to place the data on a continuation card in the
    indicated field.
```

Figure 5: The Translation Driver File TRANSLATE.DRV

12. "p" preceding a field number means that when translating TO this code it is necessary to place the data on the indicated field in the card preceding the current continuation card.
13. "s" preceding a dash means that value following the dash is to be assumed when translating FROM the particular code to another code which has a field that is not supported by the source code.
14. "a" preceding a value means that value can appear anywhere on the card. When encountered, it is copied immediately and the card is done. This is meant for the ENDT field in TABLE cards
15. A colon following a field name indicates that translation proceeds differently depending on the value that appears in the input file. The various possibilities are listed on the following lines. A * character preceding a value indicates that when translating TO the indicated program, the value shown should be substituted for the value from the source file.
16. Continuation cards are indicated by the card name preceded by a + character. On subsequent cards, field numbering begins again at 2.
17. Open-ended cards have "etc" as their last field designator.
18. The last "card name" is ENDDATA.
19. Remarks are coded at the end of this file following the flag "Remarks:'
20. Each remark consists of a single line starting with the remark number (maximum 99) followed by a period. The period is followed by a character which is either S or D or blank. If S, the remark is issued only when the indicated card is from the source file; if D, only when it is to be written in the destination file; if blank, either case. The rest of the line is the remark text.

```
------         ------         ---
ASTROS         COSMIC         MSC
version 3      1986           version 65
------         ------         ---

unknown        unknown        ACTIVE!

AEFACT         AEFACT         AEFACT
2-SID          2-SID          2-SID
3-D1           3-D1           3-D1
4-D2           4-D2           4-D2
5-D3           5-D3           5-D3
6-D4           6-D4           6-D4
7-D5           7-D5           7-D5
8-D6           8-D6           8-D6
9-D7           9-D7           9-D7
+AEFACT        +AEFACT        +AEFACT
2-D8           2-D8           2 D8
3-D9           3-D9           3-D9
etc            etc            etc
```

Figure 5: The Translation Driver File TRANSLATE.DRV (cont.)

unknown	unknown	AELIST
		2-SID
		3-E1
		4-E2
		5-E3
		6-E4
		7-E5
		8-E6
		9-E7
		+AELIST
		2-E8
		3-E9
		etc
AERO	AERO	AERO
2-ACSID	2-ACSID	2-ACSID
	3-VELOCITY	3-VELOCITY
3-REFC	4-REFC	4-REFC
4-RHOREF	5-RHOREF	5-RHOREF
	6-SYMXZ	6-SYMXZ
	7-SYMXY	7-SYMXY
AEROS(8)	unknown	AEROS(8)
2-ACSID		2-ACSID
3-RCSID		3-RCSID
4-REFC		4-REFC
5-REFB		5-REFB
6-REFS		6-REFS
7-GREF		
8-REFD		
9-REFL		
		7-SYMXZ
		8-SYMXY
unknown	unknown	AESTAT
		2-ID
		3-LABEL
ENDDATA	ENDDATA	ENDDATA

Remarks:

1.S This card should be converted to a PCOMP.
2.S PARAM values may be invalid when translated between COSMIC and MSC/NASTRAN.
3.S Transfer ASTROS DLAGS data to NASTRAN DAREA, DELAY, and DPHASE cards.
4.S Transfer NASTRAN DAREA, DELAY, and DPHASE data to ASTROS DLAGS cards.
5. SET1 cards may serve different purposes in ASTROS and NASTRAN.
6.S MSGMESH cards found. If possible, run MSC/NASTRAN with ECHO=PUNCH.
8. ASTROS AEROS cards are not entirely compatible with MSC/NASTRAN.

Figure 5: The Translation Driver File TRANSLATE.DRV (cont.)

MSC/NASTRAN but different in ASTROS; so much so that a remark is generated whenever this card type is encountered (Remark 8 at the bottom of the file).

Before any changes to TRANSNAS.DRV can take effect, a generation program called TRANSGEN must be run to convert TRANSNAS.DRV into a binary database called TRANSNAS.BIN, which is actually used by the translator in response to user requests for translation. TRANSGEN also creates a report file called TRANSGEN.RPT which summarizes its work and reports any errors.

In addition to the output bulk data file, the translator writes a report file with entries for every record that could not be translated, or for individual fields within a record.

12. Internals of FEM-X

Figure 6 illustrates the flow of data and control in FEM-X. Only the area at the center of the diagram, between the heavy vertical lines, is normally executed. This area shows FEM data files at the left, supplied by users, being read into the database and extracted (with a translation option) on the right. The functions to the left and right of the vertical lines are "system generation" functions which were discussed above.

At the top, the boxed marked "FEM-X upper layers" is the main program which interfaces with the user and the database. Below that are two databases. One is a "master" database which contains information about all systems, components, etc. In addition, for each system (as defined in section 2), a CADDB database is created; this is labelled "System database." This database is accessible to users through ICE or ASTROS as well as through FEM-X.

The rectangular box labelled "read_dat" is a Bourne shell script which processes FEM data files and enters them in a system database. The box marked "extract" is a set of Fortran routines that perform the reverse function. The "translate" box performs translation of bulk data on request.

The "read_gen" function shown on the left of the figure performs the SYSGEN process described in Section 10. It must be run separately for each of the three software formats, and there is a corresponding set of five driver files for each format. The ASTROS set of files is taken from those delivered with ASTROS, with the MAPOL sequence and error message files stripped down to reflect only the bulk data processing step.

The box labelled "transgen" is the generation of the translation database described in Section 11.

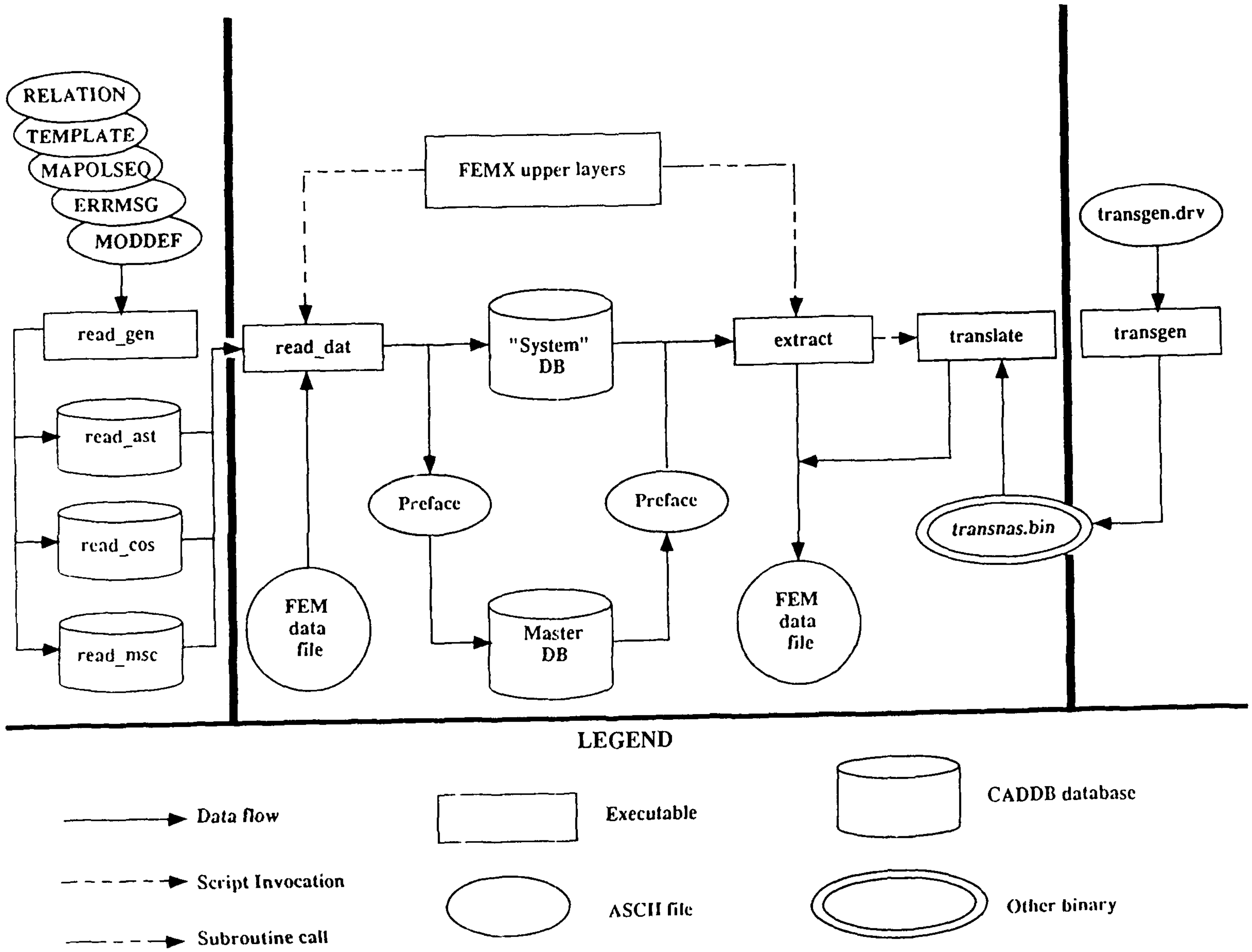


Figure 6: Flow of Control and Data in FEM-X

References

[1] Gibson, W. C., "Preservation and Utilization of Finite Element Models of USAF Aircraft Structures," CSA Report 91-01-02, January 1991.

[2] Johnson, E. H., Venkayya, V. B., Neill, D. H., Herendeen, D. L., and Hoesly, R. R., "Automated Structural Optimization System (ASTROS)," AFWAL-TR-88-3028, December 1988. Volume I – Theoretical Manual, Volume II – User's Manual, Volume III – Applications Manual, Volume IV – Programmer's Manual.

[3] Herendeen, D. L., and Ludwig, M. R., "Interactive Computer Automated Design Database (CADDB) Environment User's Manual," AFWAL-TR-88-3060, August 1988.

[4] *NASTRAN User's Manual*, NASA SP-222(08).

[5] MacNeal-Schwendler Corp., *MSC/NASTRAN User's Manual*

[6] O'Reilly & Associates, *Guides to the X Window System*, 1990.

Appendix A: Introduction to the X Window System

This Appendix is a brief introduction to the X Window System, and specifically the text editor, called `xedit`, that accompanies X, Version 11. More information on X may be found in the "X Window System User's Guide," which is Volume 3 of a series by O'Reilly & Associates [6]. Information may also be available on-line, as a `man` page.

The X Window System shares many attributes in common with other window systems such as the standard Macintosh user interface, Windows or Presentation Manager for PC's, or other workstation interfaces such as SunView. All these systems offer point-and-click operation, multiple overlapping movable windows, pull-down menus, and desk accessories like clocks and calculators. The distinguishing features of X, which have led to near universal adoption of X-based graphical user interfaces by Unix workstation vendors, are as follows:

1. It is based on a client-server computing model. "Clients" (application programs) interface with users through windows which are displayed on servers. A single server can, and usually does, display windows associated with multiple clients. Also, a client can, and often does, interact with users through multiple windows. These windows can, but rarely do, appear on different servers.

2. X is at least nominally vendor-independent.[1] While porting the server software to a new workstation is not trivial, especially if the result is to be efficient, this task has already been accomplished for virtually every kind of engineering workstation.

3. Clients can be located on computers remote from the server. X supports either TCP/IP or DecNet network protocols.

A new user of X should have `/usr/bin/X11` included in his or her path. Generally, to start X, one types `xinit`, unless that command has already been built into one's `.login` file. When `xinit` runs, it looks for a file called `.xinitrc`, and executes the commands there, which may be set up to bring up one or more windows automatically whenever X is started. If there is no `.xinitrc`, a single `xterm` window is displayed.

Windows can be manipulated using the window manager. The window manager is invoked by clicking the left mouse button anywhere in the background (i.e., any portion of the screen that is not covered by a window). A pull-down menu then appears, offering options like Resize, Move, Kill, Iconify, Raise, and Lower. When one of these options is chosen, the pointer changes shape, and the user then moves

[1]To quote the developers of the X Window System: "There is no such thing as portable software - only software that has been ported."

the pointer to the window which is to be changed. Windows can be also be resized by picking the box in the upper right corner of the window, iconified by picking the box in the upper left, and moved by grabbing the gray bar at the top and dragging.

xterm is the most common type of window. It is simply a representation of an ASCII terminal in which Unix commands may be typed and the resulting output displayed. It differs from an ASCII terminal somewhat, however. Most important, X "remembers" a certain number of lines of output that have scrolled off the top of the windows. Depending on the display options chosen, a vertical scroll bar may appear at the left of the window. By moving the scroll bar, one may recover these remembered lines. The scroll bar is manipulated by moving the mouse pointer into the scroll bar and then using the mouse buttons to move up and down. The left mouse button causes the text to scroll up, the right button causes it to scroll down, and the middle mouse button moves to a certain portion of the scroll (e.g., clicking the middle mouse button near the center of the scroll bar moves to a position near the center of the scroll. The middle button may also be used to drag the bar up and down.

`xedit` is the screen editor that comes with X11. It is based on the "Athena Text Widget," and since the text windows that appear in FEM-X are based on this same software, most of the functionality of `xedit` is also available when entering or editing data in these text windows. FEM-X users may want to use `xedit` as their primary text editor.[1]

Much of the behavior of `xedit` can be customized by users. The authors of this report have not investigated the possibilities, so the following discussion refers to the configuration of the Text Widget as it appears in FEM-X, which primarily uses default settings.[2] Also, the following discussion assumes a three-button mouse.

A "carrot" cursor appears in the window ($\wedge$) to mark the point at which text is inserted as it is typed. An I-beam cursor is controlled by the mouse. The text insertion point can be changed by moving the I-beam to the desired location and then clicking the left mouse button. It can also be changed by hitting the arrow keys, which on most keyboards appear in the right-hand keypad.

The cut and paste functions are important. In order to cut text, it must first be selected. A single word may be selected by moving the I-beam to a position within the word and double-clicking the left button. Two words may be selected by moving to the space between the words and double clicking. The entire line in which the words appear may be selected by a third click, and the entire paragraph by a fourth click. An arbitrary range of text may be selected by moving the I-beam to the front of the desired range, clicking the left mouse button, and then, while holding the left mouse button down, "dragging" to the end of the range and releasing the button.

[1]This report was created using `xedit`.

[2]Some attributes are affected by settings in the user's `.Xdefaults` file. See Volume III, *X Window System User's Guide* of the O'Reilly X Window System documentation [6] for more details.

Selected text then appears in reverse video (e.g., white on black instead of black on white). To cut the text, type ctrl-Y. Text that has been selected can be pasted by moving the insertion point to the desired location (using the mouse or the arrow keys), then clicking the middle button. Selected text can also be copied to another window (which may be an `xedit` window or some other application) by moving the cursor to that window and clicking the middle button. In this case, the original selected text does *not* disappear. It is expected that FEM-X users will use the cut-and-paste function extensively. For example, suppose a user has just finished a NASTRAN run and wants to enter information in the "Results" window of FEM-X. He could open an **`xterm`** window and view the NASTRAN output file using the Unix **`more`** command. He could cut and paste key results like eigenvalues or maximum stresses out of the NASTRAN output file and into the FEM-X "Results" window.

The search-and-replace function in `xedit` is invoked by string ctrl-S. A self-explanatory window appears in which one types the text to be searched for, and the desired replacement, if any.

Useful keystroke commands in `xedit` include:

DELETE	Delete the character to the left of the pointer
ctrl-D	Delete the character to the right of the pointer
ctrl-t	Swap the characters on either side of the pointer
ctrl-d	Delete to the end of the current word
ctrl-k	Delete to the end of the current line
ctrl-a	Move to the beginning of the current line
ctrl-e	Move to the end of the current line
meta-i	Insert a file
meta-f	Move forward one word
meta-b	Move backward one word
ctrl-v	Move forward one screenful
meta-v	Move backward one screenful
meta-shift->	Move to the end of the file
meta-shift-<	Move to the beginning of the file
ctrl-s	Bring up a search window for forward searching
ctrl-r	Bring up a search window for reverse searching
meta-q	Justify the current paragraph

(Note: on Sun workstations, the key marked "◇" functions as the meta key.)

There are three self-explanatory buttons at the top of each `xedit` window: Quit, Save, and Load. There is a vertical scrollbar and in some cases a horizontal scrollbar that are manipulated like the `xterm` scrollbar, described above.

Appendix B: Introduction to CADDB and ICE

CADDB (Computer Automated Design DataBase) was created as part of the ASTROS program. It is described in some detail in section 3.2 of Volume I of [2] and in section VIII of Volume IV. CADDB stores data in three formats: relations (tables), matrices, and unstructured entities. Matrices are stored in a manner that takes advantage of sparsity and symmetry properties and are useful for finite element matrix operations. Unstructured entities are for scratch data. Relations are the most interesting entities in CADDB databases. Relations have "attributes" which can be thought of as column headings in tables. Each column or attribute can store either integer, real, or character data. CADDB provides "projections" or constraints that can be specified when fetching records from a relation. One can specify the particular columns to be fetched, and also specify complex conditions which can be applied to the selection process. Examples of these conditions are given below.

CADDB databases are accessible in three ways: to users through ICE, to Fortran programmers through a subroutine library, and to MAPOL programmers. MAPOL is the matrix manipulation language that ASTROS implements. Programmers can use CADDB for purposes unrelated to ASTROS or even unrelated to engineering computation. ICE (Interactive CADDB Environment) is an interactive program that provides users with access to CADDB databases. Its original purpose was to allow users to view and manipulate results generated by ASTROS, which are stored in the database. It may also be useful to users of FEM-X, who can access bulk data databases that are created by FEM-X.[1] ICE is described in [3]. A brief introduction to ICE follows.

ICE is invoked with a command line, just `ice` on most systems. Thereafter, ICE provides a prompt (`>ICE`) and awaits user commands. All ICE commands end with a semicolon, which is easy to forget. If it is forgotten, ICE will issue a prompt for a continuation of the command, at which point the semi-colon can be typed.

The first ICE command is almost always a request to open a database. Only one database can be open at any one time. CADDB databases are stored as a pair of files with extensions `.D01` and `.IDX`. The database name given to ICE corresponds to the first part of these file names (e.g., a database named `WING` would be stored in `WING.D01` and `WING.IDX`). A command like

```
OPEN WING;
```

is issued to open a database. At this point, ICE prompts for the password associated with the database, which is not echoed when typed. All databases created by FEM-X have the password `FEMX`. Databases created by ASTROS have their password entered on the `ASSIGN DATABASE` line in the ASTROS input file.

[1] It is also possible to access the master FEM-X database in this manner, but this is not recommended.

When inspecting a database, a user might typically start with a DESCRIBE command which lists all the entities (relations, matrices, and unstructured entities) that are present in the database. Then one could ask for information about a particular entity by typing, for example, DESCRIBE GRID. This would list all the attribute (column) names in the relation called GRID, and how many entries there were currently.

Next, one could see all the data in the GRID relation by typing **SELECT * FROM GRID;**. The result would likely be a screen full of data like that shown in the first panel of Figure 7. The headers show the names of the attributes of the GRID relation: ID,CP,X,Y,Z,CD,PSPC,SEID, and the rest of the screen shows the first few entries in the relation. This display is rather cluttered, so one might type **x** at the prompt and revise the command to show only the interesting attributes:

```
SELECT ID,X,Y,Z FROM GRID;
```

The result would be as shown in the second panel of the figure. Hitting the RETURN key continually would bring up more screens full of data. If one wanted to restrict the selection of records, one could type

```
SELECT ID,X,Y,Z FROM GRID WHERE Y>600 AND Y<602 AND ID<50000;
```

(see the third panel). Much more complexity is possible; see [3] for details.

Rather than typing interactive commands, users may prepare commands in advance and enter them into a file using a text editor. For example, a script file named F18.SCRIPT could be played through ICE by typing

```
SET SCRIPT TO 'F18.SCRIPT';
```

and all the commands in that file would be executed serially.

Data can be taken from ICE and written to a file. This is done with the **SET INTERFACE** and **INTERFACE FORMAT** commands. For example, the sequence

```
SET INTERFACE TO 'GRID.BULK';
INTERFACE FORMAT '(''GRID'',4X,2I8,3F8.3,I8)';
INTERFACE ON;
SELECT ID,CP,X,Y,Z,CD FROM GRID
   WHERE Y>600 AND Y<602 AND ID<50000;
```

would write out the selected GRID records in NASTRAN bulk data format.

```
ICE> select id,x,y,z from grid;
   ID           X            Y            Z
-----  ------------ ------------ ------------
   13  -4.55000E+00 -4.55000E+00  0.00000E+00
   14  -2.00000E+00 -4.55000E+00  0.00000E+00
   15   0.00000E+00 -4.55000E+00  0.00000E+00
   16   2.00000E+00 -4.55000E+00  0.00000E+00
   17   4.55000E+00 -4.55000E+00  0.00000E+00
   23  -4.22500E+00 -4.22500E+00  0.00000E+00
   24  -2.00000E+00 -4.22500E+00  0.00000E+00
   25   0.00000E+00 -4.22500E+00  0.00000E+00
   26   2.00000E+00 -4.22500E+00  0.00000E+00
   27   4.22500E+00 -4.22500E+00  0.00000E+00
   33  -3.67500E+00 -3.67500E+00  0.00000E+00
   34  -2.00000E+00 -3.67500E+00  0.00000E+00
   35   0.00000E+00 -3.67500E+00  0.00000E+00
   36   2.00000E+00 -3.67500E+00  0.00000E+00
   37   3.67500E+00 -3.67500E+00  0.00000E+00
   41  -4.55000E+00 -2.00000E+00  0.00000E+00
   42  -4.22500E+00 -2.00000E+00  0.00000E+00
   43  -3.67500E+00 -2.00000E+00  0.00000E+00
   44  -2.00000E+00 -2.00000E+00  0.00000E+00
   45   0.00000E+00 -2.00000E+00  0.00000E+00
   46   2.00000E+00 -2.00000E+00  0.00000E+00
ICE> Press the Space bar or the Return key to continue:
```

```
ICE> select * from grid;
   ID    CP            X            Y            Z           CD  PSPC
-----  -----  ------------ ------------ ------------ ------------ -----
 SEID
-----
   13      0  -4.55000E+00 -4.55000E+00  0.00000E+00  0.00000E+00     0
    0
   14      0  -2.00000E+00 -4.55000E+00  0.00000E+00  0.00000E+00     0
    0
   15      0   0.00000E+00 -4.55000E+00  0.00000E+00  0.00000E+00     0
    0
   16      0   2.00000E+00 -4.55000E+00  0.00000E+00  0.00000E+00     0
    0
   17      0   4.55000E+00 -4.55000E+00  0.00000E+00  0.00000E+00     0
    0
   23      0  -4.22500E+00 -4.22500E+00  0.00000E+00  0.00000E+00     0
    0
   24      0  -2.00000E+00 -4.22500E+00  0.00000E+00  0.00000E+00     0
    0
   25      0   0.00000E+00 -4.22500E+00  0.00000E+00  0.00000E+00     0
    0
   26      0   2.00000E+00 -4.22500E+00  0.00000E+00  0.00000E+00     0
    0

ICE> Press the Space bar or the Return key to continue:
```

```
ICE> select id,x,y,z from grid where y=4.55;
   ID           X            Y            Z
-----  ------------ ------------ ------------
   93  -4.55000E+00  4.55000E+00  0.00000E+00
   94  -2.00000E+00  4.55000E+00  0.00000E+00
   95   0.00000E+00  4.55000E+00  0.00000E+00
   96   2.00000E+00  4.55000E+00  0.00000E+00
   97   4.55000E+00  4.55000E+00  0.00000E+00
  252  -3.73333E+00  4.55000E+00  0.00000E+00
  253  -2.86667E+00  4.55000E+00  0.00000E+00
  254  -1.17500E+00  4.55000E+00  0.00000E+00
  255  -6.66670E-01  4.55000E+00  0.00000E+00
  256   6.66667E-01  4.55000E+00  0.00000E+00
  257   1.17500E+00  4.55000E+00  0.00000E+00
  258   2.86667E+00  4.55000E+00  0.00000E+00
  259   3.73333E+00  4.55000E+00  0.00000E+00
.... 13 ENTRIES SELECTED
ICE>
ICE>
```

Figure 7: Sample ICE Output.

Appendix C: List of Files

The following files are delivered:

SYSGEN files for COSMIC NASTRAN and MSC/NASTRAN data:[1]

READ_COS.MAPOLSEQ	READ_MSC.MODDEF
READ_COS.MODDEF	READ_MSC.RELATION
READ_COS.RELATION	READ_MSC.SERRMSG
READ_COS.SERRMSG	READ_MSC.TEMPLATE
READ_COS.TEMPLATE	READ_MSC.MAPOLSEQ

Main programs for rump ASTROS versions, adapted from `astros.f`, supplied with ASTROS; executables, and system database files:

read_ast.for	read_cos.for	read_msc.for
read_ast	read_cos	read_msc
READ_AST.D01	READ_COS.D01	READ_MSC.D01
READ_AST.IDX	READ_COS.IDX	READ_MSC.IDX

Code to generate an alternate title page (MAPOL modules):

astttl.for	costtl.for	mscttl.for

Code to read a bulk data template file and compile a list of all possible record types (run only when templates are modified) – source and executable.

carddef1.for	carddef1

List of card types, each created by one execution of `carddef1`.

CARDAST.DEF	CARDCOS.DEF	CARDMSC.DEF

Code that reads card type files (`CARDxxx.DEF`) and writes the list of card types into relation CARDDEF in the user database (run each time a bulk data file is processed)

carddef2.for	carddef2

Bourne shell script for reading bulk data into a database:

read_dat

Translation template, binary database, and generation report:

[1]SYSGEN files for ASTROS are delivered with ASTROS.

TRANSGEN.DRV TRANSGEN.BIN TRANSGEN.RPT

Translation source code:

dtranslate.for translate.for

Driver and executable for stand-alone translator:

dtrans.for dtrans

Fortran subroutine for bulk data extraction:

extract.for

Other Fortran source code and common blocks:

*.for *.cmn

Other C source code and "include" files:

*.c *.h

"Makefile" for compilation, linking, and system generation:

Makefile

Files containing "help" text and error message text:

help-data notice-data

U.S. Government Printing Office 548-076